REMARKS

Status of the Claims

Claims 1-20 are currently present in the Application, and claims 1, 8, and 14 are independent claims. Claims 1, 8, and 14 have been amended, no claims have been cancelled, and no claims have been added in this response.

Examiner Interview

Applicants note with appreciation the telephonic interview conducted between Applicants' representative and the Examiner on November 20, 2006. During the telephonic interview, the Examiner and Applicants' representative discussed the 102 reference (Akgul et al., U.S. Patent Pub. 2003/0074650). In particular, Applicants' representative discussed that Applicants' invention first initiates a debugger thread, and then invokes an operational thread using the debugger thread. In contrast, Akgul's operational thread invokes a debugger thread when an error occurs during application code execution, which is opposite of that claimed by Applicants. The Examiner suggested that since Akgul's claim 8 limitation states the debugger system comprises a "breakpoint or a single stepping support," that Akgul's debugger thread may invoke Akgul's operational thread.

Regarding Applicants' claim 5 limitations, Applicants' representative discussed that Applicants' invention loads a debugger operating system on the debugger thread and loads a primary operating system on the operational thread. In contrast, Akgul teaches a single operating system that supports both an operational thread and a debugger thread. The Examiner understood Applicants' representative's position, and wished to view Akgul in more detail. No agreement was reached regarding the claims.

Drawings

Applicants note that the Examiner did not indicate whether the formal drawings, filed with Applicants' application, are accepted by the Examiner. Applicants respectfully

request that the Examiner indicate whether the formal drawings are accepted in the next office communication.

Claim Rejections - Alleged Anticipation Under 35 U.S.C. § 102

Claims 1-20 stand rejected under 35 U.S.C. § 102(e) as being anticipated by Akgul et al. (U.S. Patent Pub. 2003/0074650, hereinafter "Akgul"). Applicants respectfully traverse these rejections.

Applicants' have amended claim 1's second limitation to distinctly claim that Applicants' operational thread starts, for the first time, when invoked by Applicants' debugger thread. Support for this amendment can be found in Applicants' Figure 2, reference 215, and corresponding text, which shows that Applicants' operational thread does not exist prior to being invoked by Applicants' debugger thread. Therefore, no new matter is added with such amendment. As amended, claim 1 is directed towards a method of debugging software code with limitations comprising:

- initiating a debugger thread on a computer system, wherein the debugger thread performs a plurality of debugger events;
- invoking an operational thread on the computer system using the debugger thread, the operational thread performing operational tasks, wherein the invoking starts the operational thread for a first time;
- executing the software code using the operational thread;
- detecting whether the operational thread is functioning using the debugger thread; and
- debugging the operational thread using the debugger thread in response to the detecting.

Applicants' invention loads a debugger thread, and then invokes an operational thread using the debugger thread. The reason that the debugger thread invokes the operational thread is that Applicants' invention is able to debug new, unstable operating systems running on the operational thread because the debugger thread invoked the operational thread (see below).

In contrast, Akgul's operating system thread actually invokes Akgul's debugger thread, which is opposite of that claimed by Applicants. In fact, Akgul teaches away from initiating a debugger thread altogether unless an error occurs during application code execution in order to conserve resources. Particularly, Akgul states:

> "During error-free operation, at runtime, the **debugger modules are not typically loaded** into the system provided that the user does not request any manual module loading. This prevents consumption of extra system resources by the debugging software. When the debugger modules are not loaded into the system, they are kept generally in a storage unit external to the target system...**As soon as an error condition occurs...the debugger modules are loaded** from the external storage unit into the memory of the target system and are dynamically linked to the OS." (page 5, para. 44, emphasis added)

As can be seen from the above excerpt, Akgul loads debugger modules using an operational thread when an error occurs, which is opposite of Applicants' invention. During the Examiner interview, the Examiner suggested that since Akgul's claim 8 includes a debugger system limitation of a "break point or a single stepping support," that Akgul teaches a debugger thread invoking an operational thread. There is a difference, however, in what Applicants' claim and what Akgul teaches. Applicants' claim that the debugger thread "*invokes an operational thread on the computer system using the debugger thread...* ***wherein the invoking starts the operational thread for a first time.***" Meaning, the operational thread does not exist until invoked by the debugger thread. Akgul teaches that the operational thread actually invokes the debugger thread, which means that Akgul's operational thread has to be running prior to invoking Akgul's debugger thread.

Therefore, since Akgul does not teach or suggest all the limitations included in Applicants' claim 1 as amended, amended claim 1 is allowable over Akgul. Claim 8 is an information handling system claim including similar limitations as claim 1 and, therefore, is allowable for at least the same reasons as claim 1 is allowable. Claim 14 is a computer program product claim including similar limitations as claim 1 and, therefore, is allowable for at least the same reasons as claim 1 is allowable.

Notwithstanding the fact that claim 5 is dependent upon claim 1 and, therefore, allowable for at leas the same reasons as claim 1, claim 5 adds limitations to claim 1 of:

- loading a debugger operating system on the debugger thread; and
- loading a primary operating system on the operational thread, wherein the debugger operating system is different from the primary operating system.

Applicants' invention loads a debugger operating system on a debugger thread and loads a primary operating system on an operational thread. In turn, Applicants' debugger thread detects whether the primary operating system running on the operational thread is functioning correctly, and debugs the operational thread accordingly. In contrast, Akgul discloses a single operating system from which all threads execute, regardless of whether they are operational threads or debugger threads. Particularly, Akgul states:

> "In the exemplary embodiment, software debugging features are distributed between and among distinct debugger modules that are compiled independently from each other. The user can select the necessary modules corresponding to the desired debugging features at compile time. In addition to compile time selection, **each module is a dynamically loadable and linkable part of the OS**. Advantageously, when a debugger module is needed at runtime for extra debugging features, that module can be added to the system without the requirement of application or OS recompilation or a reboot." (page 4, para. 35, emphasis added)

As can be seen from the above excerpt, Akgul's debugger modules function under a primary operating system and do not use a separate operating system as claimed by Applicants. Again, the reason being that Applicants' invention is able to debug new, unstable operating systems whereas Akgul debugs application code using a stable operating system.

Therefore, since Akgul does not teach or suggest all the limitations included in Applicants' claim 5, claim 5 is allowable over Akgul. Claim 12 is an information handling system claim including similar limitations as claim 5 and, therefore, is allowable for at least the same reasons as claim 5 is allowable. Claim 18 is a computer program

product claim including similar limitations as claim 5 and, therefore, is allowable for at least the same reasons as claim 5 is allowable.

Each of the remaining claims 2-4, 6-7, 9-11, 13, 15-17, and 19-20, each depend, either directly or indirectly, upon allowable independent claims 1, 8, or 14. Therefore, claims 2-4, 6-7, 9-11, 13, 15-17, and 19-20 are also allowable for at least the same reasons as their respective independent claims are allowable.

Conclusion

As a result of the foregoing, it is asserted by Applicants that the remaining claims in the Application are in condition for allowance, and Applicans respectfully request an early allowance of such claims.

Applicants respectfully request that the Examiner contact the Applicants' attorney listed below if the Examiner believes that such a discussion would be helpful in resolving any remaining questions or issues related to this Application.

Respectfully submitted,

By /Leslie A. Van Leeuwen, Reg. No. 42,196/

Leslie A. Van Leeuwen, Reg. No. 42,196
Van Leeuwen & Van Leeuwen
Attorney for Applicants
Telephone: (512) 301-6738
Facsimile: (512) 301-6742